1. **Describe UML class diagram and its purpose.**

Unified Modelling Language (UML) is a general purpose visual modelling language that helps to visualize, construct, and document software systems by showing the system’s classes, their attributes, operations, and the relationships among objects. It uses graphic notation to create visual models of software systems.

Its purpose:

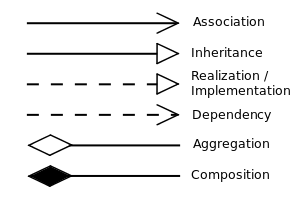
* To model the static view of an application
* For constructing executable code of the software
* Describe responsibilities of a system

1. **List and describe types of class relationships in OOP. Also, draw their UML symbol.**

Classes are interrelated to each other in specific ways. In particular, relationships in class diagrams include different types of logical connections. The following are logical connections used as class relationships in [UML](https://creately.com/lp/uml-diagram-tool):

* **Association** - It is a structural relationship that represents objects can be connected or associated with another object inside the system. It encompasses just about any logical connection or relationship between classes.
* **Directed Association** - is related to the direction of flow within association classes. In a directed association, the flow is directed, that is the association from one class to another class flows in a single direction only.
* **Reflexive Association** - this type of association is for a class that has multiple functions or instances that interact with each other.
* **Inheritance / Generalization** – also called parent-child relationship, exists between a **superclass (parent class) and a subclass (child class)** such that the subclass inherits data structures as well as operation definitions from the superclass without the need to define them again. In general it allows us to define new classes (subclass) from existing classes (superclass).
* **Realization / Implementation** -is a relationship between two objects, where one model element implements the responsibility specified by another model element which means one entity denotes some responsibility which is not implemented by itself and the other entity that implements them.
* **Aggregation** -more specialized version of the association relationship that **an object of one class can own or access the objects of another class or it represents an ownership relationship between two objects**.
* **Composition** - is a restricted form of Aggregation that represents part-of relationship in which two entities are highly dependent on each other. It is a relationship in which a composite object consists of non-shareable objects, and the latter are existence-dependent to the former.
* **Dependency-** is a relationship that states that one class uses the information and services of another class, but not necessarily the reverse. It can relate how various things inside a particular system are dependent on each other. Dependency is used to describe the relationship between various elements in UML that are dependent upon each other.
* **Multiplicity** -is the active logical association when the cardinality of a class in relation to another is being depicted. It is an indication of how many objects may participate in a given relationship or the allowable number of instances of the element.

**UML relationship symbols**

****